REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From – To)*
27-11-2006	Final Report	29 November 2005 - 29-Jan-07

4. TITLE AND SUBTITLE

Dynamic Testing and Automatic Repair of Reconfigurable Wiring Harnesses

5a. CONTRACT NUMBER
FA8655-06-1-3021

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Sarah Thompson, Alan Mycroft

5d. PROJECT NUMBER

5d. TASK NUMBER

5e. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

University of Cambridge
William Gates Building
Cambridge CB3 0FD
United Kingdom

8. PERFORMING ORGANIZATION REPORT NUMBER

N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

EOARD
PSC 821 BOX 14
FPO AE 09421-0014

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
Grant 06-3021

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Spacecraft design is, without doubt, one of the most challenging areas of modern engineering. In order to be viable, spacecraft must mass relatively little, whilst being capable of surviving the considerable G-forces and vibration of launch. In space, they must withstand extreme temperatures, hard vacuum and high levels of radiation, for several years without maintenance.

Conventionally, spacecraft wiring harnesses are built with architectures that are fixed at the time of manufacture. They must therefore be designed to endure the lifetime of the mission with a very high probability, though the conventionally necessary redundant duplication of signals has significant implications for mass. Given that launch costs are typically in excess of $30,000 per kg, reducing the mass of a spacecraft's wiring harness, without compromising reliability, is highly desirable. As a motivating example, the network cabling in the International Space Station (ISS) is known to mass more than 10 metric tonnes.

Recent advances in MEMS-based switching have made it possible to consider the construction of reconfigurable manifolds – essentially, wiring harnesses that behave like macroscopic FPGA routing networks. Redundant wiring can be shared between many signals, thereby significantly reducing the total amount of cable required. Reconfigurability has a significant further benefit, in that it also allows adaptation to mission requirements that change over time, whilst also significantly reducing design time.

In a recent initiative, the US Air Force has been moving toward a responsive space paradigm which aims to reduce the time from design concept to launch (currently several years) to less than one week. Such a target is unlikely to be achievable with existing bespoke one-off design techniques; a parts-bin driven, plug-and-play approach to satellite construction will become essential. It must be possible to choose a satellite chassis of a size appropriate to the task in terms of accommodating sufficient manoevering propellant as well as the necessary instrumentation payload, then bolt everything together and have the resulting satellite 'just work.'

We present an approach that allows such a reconfigurable manifold to be automatically self-configured, then dynamically tested in-situ, such that signals are automatically rerouted around non-functioning wires and switches as soon as faults are detected. Make-before-break switching is used in order to allow wires to that are currently in use to be rerouted transparently from the point of view of subsystems that are interconnected by the manifold, whilst also making it possible to achieve near-100% testability.

15. SUBJECT TERMS
Electronic Devices, Satellite damage, MEMS, Space vehicles, FPGA, Computational Mathematics, EOARD

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18, NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			BARRETT A. FLAKE
UNCLAS	UNCLAS	UNCLAS	UL	30	19b. TELEPHONE NUMBER *(Include area code)* +44 (0)20 7514 4285

Reconfigurable Manifolds Final Report

Sarah Thompson **Alan Mycroft (PI)**

{sarah.thompson,alan.mycroft}@cl.cam.ac.uk

Computer Laboratory, University of Cambridge, William Gates Building, 15 JJ Thomson Ave., Cambridge CB3 0FD, UK

1 Introduction

Spacecraft design is, without doubt, one of the most challenging areas of modern engineering. In order to be viable, spacecraft must mass relatively little, whilst being capable of surviving the considerable G-forces and vibration of launch. In space, they must withstand extreme temperatures, hard vacuum and high levels of radiation, for several years without maintenance.

Conventionally, spacecraft wiring harnesses are built with architectures that are fixed at the time of manufacture. They must therefore be designed to endure the lifetime of the mission with a very high probability, though the conventionally necessary redundant duplication of signals has significant implications for mass. Given that launch costs are typically in excess of $30,000 per kg, reducing the mass of a spacecraft's wiring harness, without compromising reliability, is highly desirable. As a motivating example, the network cabling in the International Space Station (ISS) is known to mass more than 10 metric tonnes.

Recent advances in MEMS-based switching [25] have made it possible to consider the construction of *reconfigurable manifolds* – essentially, wiring harnesses that behave like macroscopic FPGA routing networks. Redundant wiring can be shared between many signals, thereby significantly reducing the total amount of cable required. Reconfigurability has a significant further benefit, in that it also allows adaptation to mission requirements that change over time, whilst also significantly reducing design time.

In a recent initiative, the US Air Force has been moving toward a *responsive space* paradigm which aims to reduce the time from design concept to launch (currently several years) to less than one week [17]. Such a target is unlikely to be achievable with existing bespoke one-off design techniques; a parts-bin driven, plug-and-play approach to satellite construction will become essential. It must be possible to choose a satellite chassis of a size appropriate to the task in terms of accommodating sufficient manoevering propellant as well as the necessary instrumentation payload, then bolt everything together and have the resulting satellite 'just work.'

We present an approach that allows such a reconfigurable manifold to be automatically self-configured, then dynamically tested in-situ, such that signals are automatically rerouted around non-functioning wires and switches as soon as faults are detected. Make-before-break switching is used in order to allow wires to that are currently in use to be rerouted transparently from the point of view of subsystems that are interconnected by the manifold, whilst also making it possible to achieve near-100% testability.

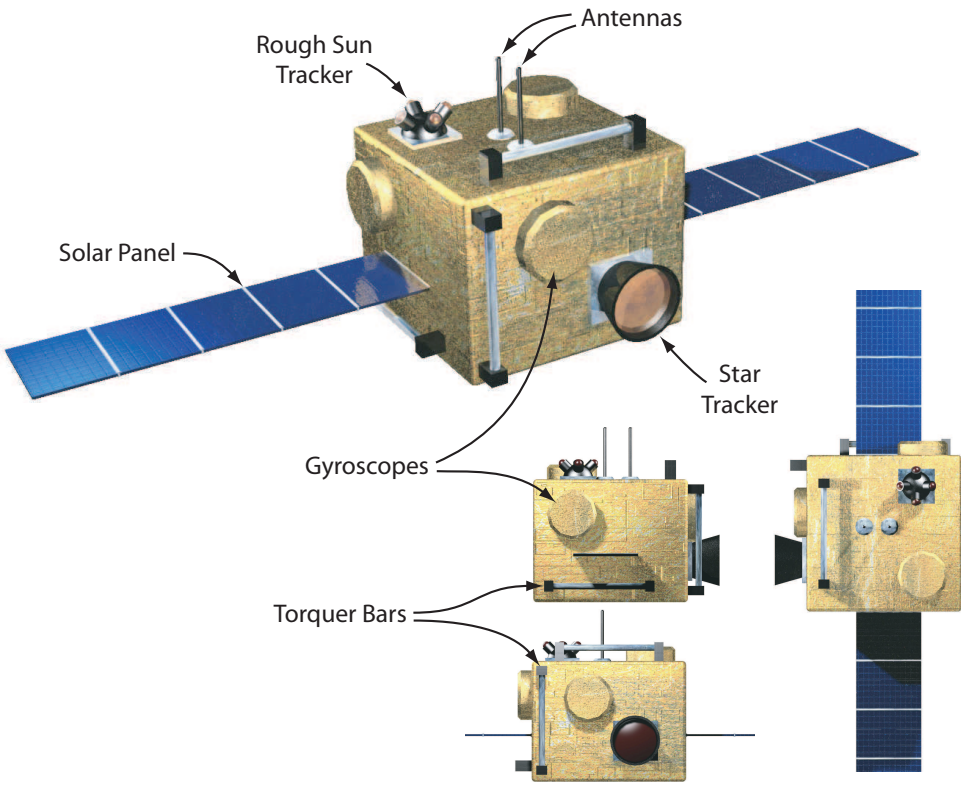

Figure 1: A typical near-earth small satellite configuration

1.1 Physical satellite wiring architectures

Conventionally, satellites are constructed with fixed wiring architectures. Reliability must therefore be engineered-in through modular redundancy – duplication or triplication (or more) of signal paths is common, which carries with it an attendant mass penalty.

Typically, two kinds of wiring architecture are common:

Card frame with passive backplane Fig. 2 shows a typical passive backplane with multiple subsystems, each slotting in to a rack on separate cards[1].

Motherboard/daughterboard Another common approach is shown in Fig. 3, where a single motherboard has a number of daughter boards attached to it on standoffs. Normally (though not visible in the diagram) these daughter boards plug directly into connectors on the motherboard, again avoiding the need for cables.

Wiring harnesses, in the sense that they exist in cars and aircraft as bundles of physical cables, tend to be avoided where possible because of their greater mass and poorer reliability.

Typically, card frames have passive backplanes, which do not normally contain active electronics beyond perhaps some simple power regulation or line termination. Motherboard approaches more commonly include active electronics on the main board itself, though this is not a prerequisite.

[1]Note that the image is representational – actual satellite hardware differs in detail

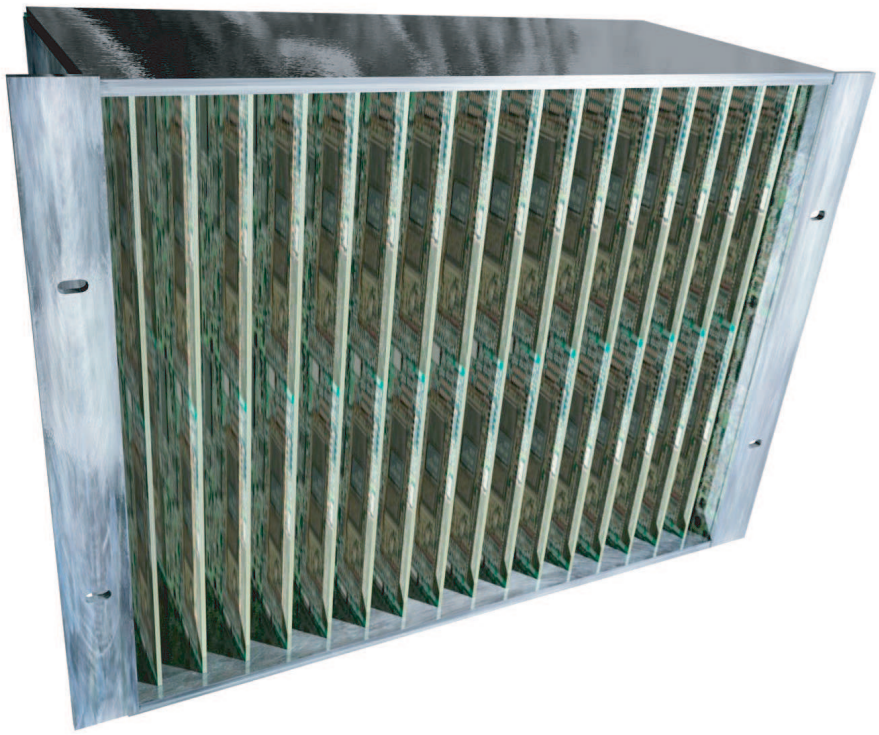

Figure 2: Card frame with backplane

1.2 Logical satellite wiring architectures

At a logical, block diagram level, fixed architecture satellite wiring harnesses typically follow the structure shown in Fig. 4. All of the main subsystems are attached to a motherboard or backplane that provides most of the necessary interconnection infrastructure, with external devices plugging directly into the relevant subsystems. All required redundancy must be in place from the outset. Typically, satellites are one-off designs, so any design changes before launch require physical modifications – of course, such changes *after* launch are typically impossible. As a further consequence of this approach, subsystem re-use is relatively uncommon, requiring considerable effort in terms of design, validation and verification, of the order of several years from concept to launch.

2 Reconfigurable manifolds

The responsive space paradigm [17] implies the requirement to move away from fixed architectures and their consequential design and validation costs toward an autonomous, self-organising approach. In essence, a *reconfigurable manifold* is a self-organising, self-testing, self-repairing replacement for a fixed architecture wiring harness. Ideally, at a system level, a spacecraft adopting this approach should have an architecture similar to that shown in Fig. 5.

Ideally, all wiring should be routed by the manifold rather than connected directly to subsystems. From a the point of view of rapid construction, this is ideal – a subsystem

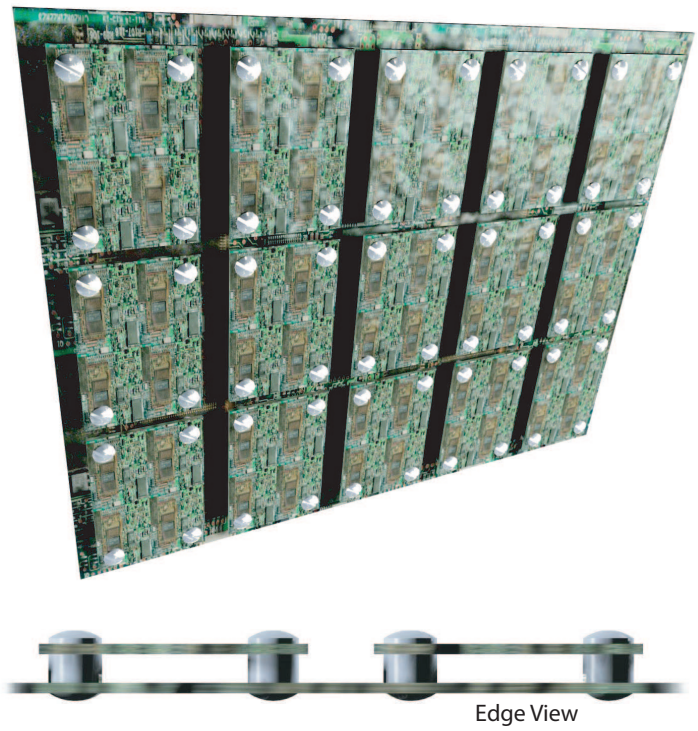

Figure 3: Motherboard with attached daughter boards

such as a gyroscope, star tracker, sun tracker or antenna could be bolted to the spacecraft chassis anywhere that is physically convenient, with all of the necessary wiring being 'discovered' and automatically routed after power-up.

2.1 Signal types

Spacecraft wiring harnesses (reconfigurable or otherwise) must be able to carry a wide variety of signals, varying in terms of power, voltage and bandwidth, with similarly variable electrical considerations in terms of impedance, end-to-end resistance, etc. Typical signal types found in satellites, along with example applications are listed as follows:

Power Normally a single $+28$V DC unregulated supply rail powers the entire spacecraft, with local step-down regulators providing lower voltage high quality supply rails to each subsystem. Where higher voltages are necessary, e.g. to drive cryocoolers for low background noise imaging sensors, this is normally achieved with local step-up switching DC-DC converters.

Heavy current analogue High current feeds to torquer bars, motor drives, solenoid power, explosive bolts, etc.

Low current, low speed analogue Analogue sensor feeds, thermocouples, rough sun tracker photocells, etc.

Low current, high speed analogue Higher speed sensor wiring, video feeds from cameras and star trackers, etc.

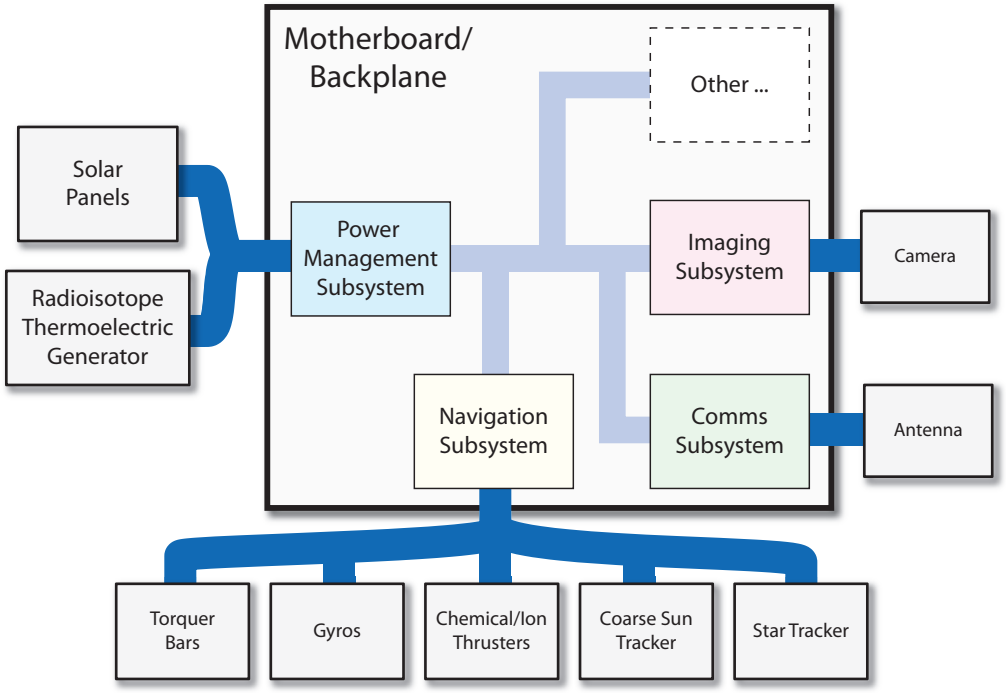

Figure 4: Typical non-reconfigurable satellite wiring architecture

Low speed digital Simple on/off telemetry sensors, e.g. mechanical limit switches.

High speed digital Digital communications between subsystems.

Low power microwave Radio receiver antenna feeds, low power radio transmitter antenna feeds.

High power microwave High power antenna feeds, ion thruster power cabling, etc.

Optical High speed network connectivity, lower speed sensor applications that require a significant degree of electrical isolation[2].

No single switching architecture, at the time of writing, can accommodate more than a few of the above signal types.

2.2 Constructing practical reconfigurable manifolds

A practical reconfigurable manifold must encompass most, if not all, signal types in order to be effective. Since no single switch fabric is suitable, it makes sense to split the manifold into separate sub-manifolds, each of handling a different signal type, as shown in Fig. 6.

[2]Optical switching is beyond the scope of this work and will not be discussed further.

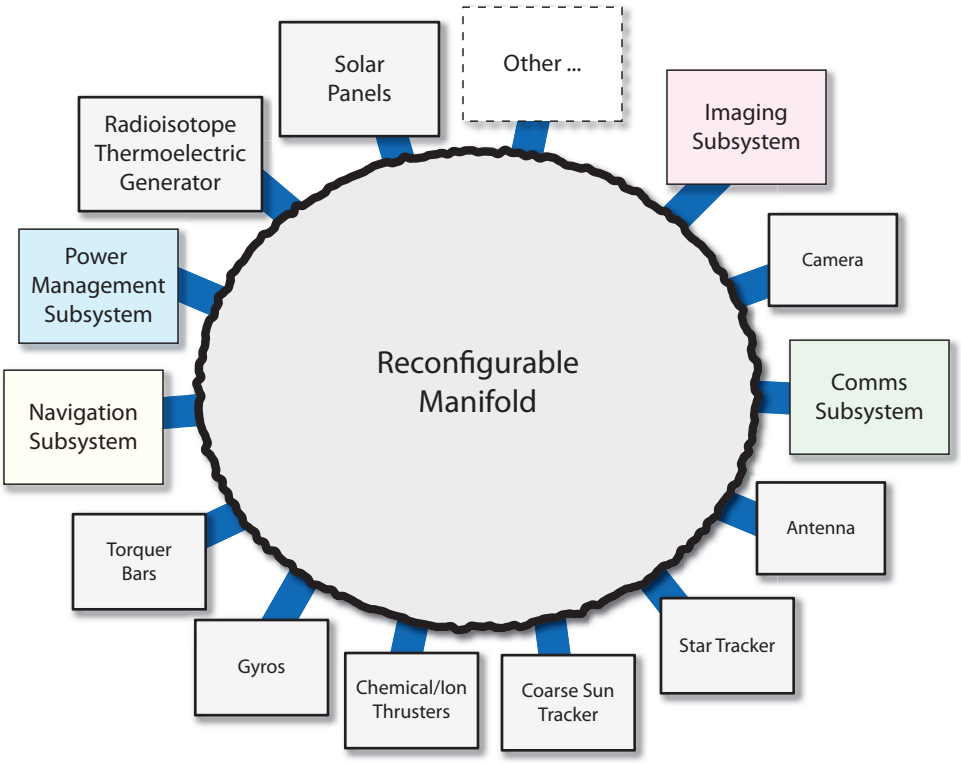

Figure 5: Reconfigurable manifold architecture

Some cross-connectivity between the sub-manifolds makes sense, since, for example, several MEMS relays could potentially be connected in parallel in order to switch heavier current, or DC-biased analogue routing with sufficient bandwidth could, in an emergency on orbit, be used to carry digital data.

Fig. 7 shows a reconfigurable manifold implemented as a replacement for a passive backplane or passive motherboard. In contrast with Fig. 4, external systems connect to the manifold rather than direct to the subsystems themselves. Configuring such a satellite might be as simple as installing cards in a backplane or motherboard in any convenient order, then plugging external devices into the manifold. Spare slots could, given sufficient mass budget, be used to provide extra redundancy simply by plugging in extra duplicate cards; appropriate firmware could potentially handle this automatically.

An alternative architecture is shown in Fig. 8. Rather than a single manifold routing between devices connected to its periphery, the manifold is itself distributed between the subsystems. Interconnection between subsystems is passive, with the subsystems cooperating to establish longer distance, multi-hop routes.

The single manifold approach is perhaps best suited to small satellites, whereas the (more complex, though more flexible and scalable) distributed approach lends itself to larger spacecraft such as large satellites, manned spacecraft, space stations or indeed also to terrestrial aircraft.

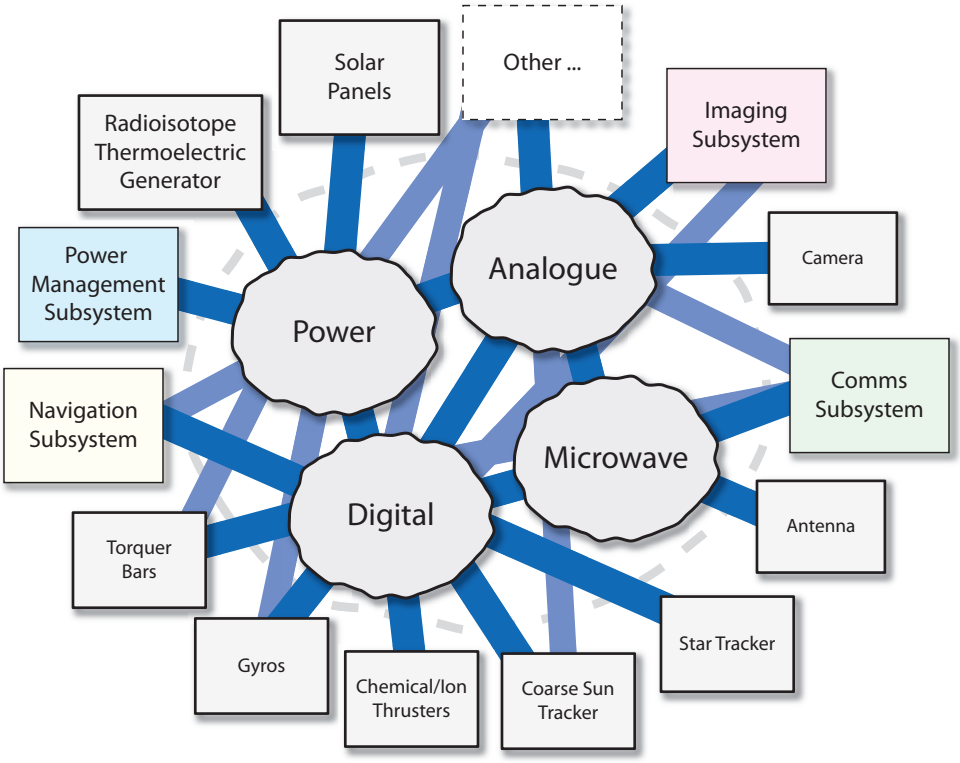

Figure 6: Separate routing networks for power, analogue, digital and microwave

2.3 Switching technologies

Many switching technologies exist that differ considerably in capability:

FPGAs Field-programmable gate arrays can be used to route digital data, and are also comparatively cheap and readily available.

FPTAs Field-programmable transistor arrays [31] have some similarities to FPGAs, though they are aimed more closely at analogue applications. As with FPGAs, they are not intended from the outset as routing devices for use within a the switch fabric of a reconfigurable manifold, though it would seem feasible to apply them to the switching of low- to medium-speed analogue signals.

Digital Crossbar Switch ASICs A number of commercial, off-the-shelf (COTS) digital crossbar switch chips are available, though this application appears to be becoming dominated by FPGAs as a consequence of the larger FPGA manufacturers getting more directly involved by releasing support for using their devices in this way [4].

Analogue Crossbar Switch ASICs Though not so widely supported as digital crossbar

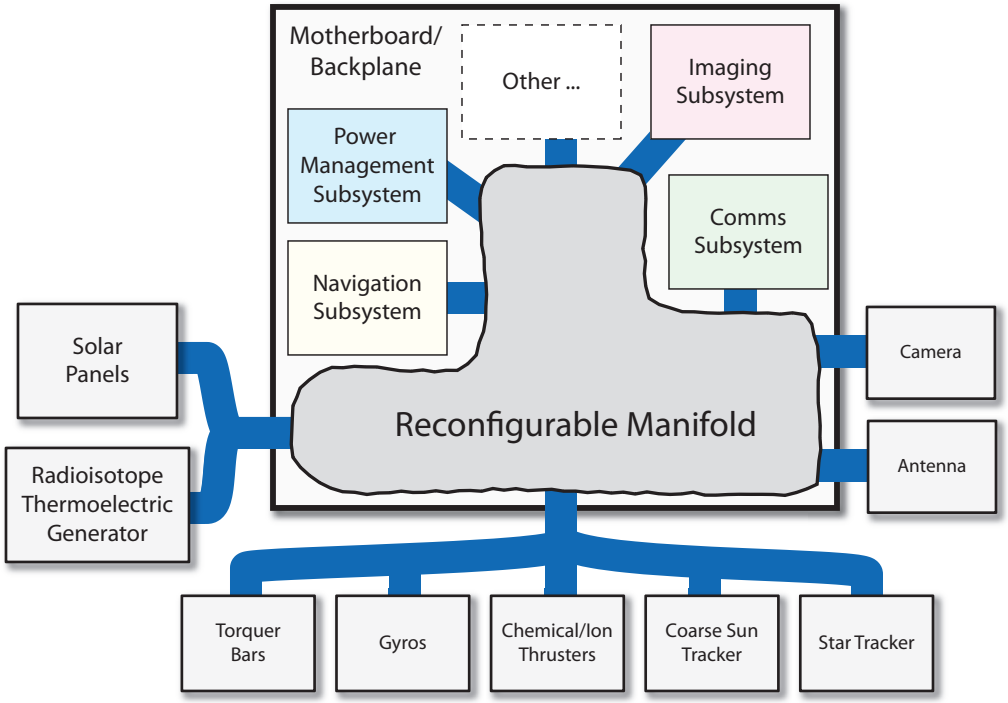

Figure 7: Reconfigurable manifold as a motherboard or backplane

switch devices, analogue crossbar switches are available, mostly aimed at switching analogue video signals [3].

MEMS switches Micron-scale electromechanical switches have been demonstrated to be an effective candidate technology [25]. Though physically far larger than CMOS transistor-based switches, MEMS switches are nevertheless orders of magnitude smaller and lighter than full-size mechanical relays, and have excellent electrical characteristics that renders them capable of being applied to almost any low-current switching application, including microwave.

Electromechanical Relays Somewhat old-fashioned, relays are nevertheless capable of switching very heavy currents. They are sufficiently massive, however, that it is difficult to imagine them being used in large numbers in a spacecraft application.

Discrete MOSFET/IGBT Switching Large power transistors, both MOS and bipolar, are commonly used to switch heavy current and moderately high voltage (up to a few hundred volts and/or hundreds of amps) signals, particularly in motor drive applications. They exhibit high reliability and relatively good radiation hardness characteristics due to their very large (in comparison with ASICs) geometries, though their gate drive circuitry can be tricky to engineer. Though physically bulky, they nevertheless remain a useful possibility for constructing heavy current and/or power switching networks.

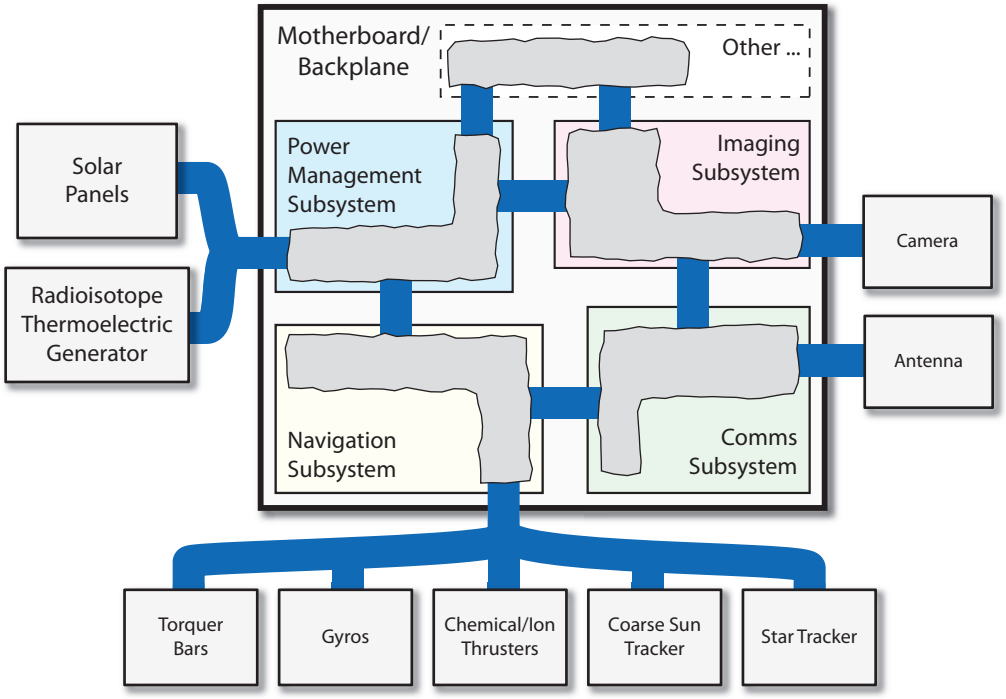

Figure 8: Reconfigurable manifold distributed across subsystems

Table 1 shows compatibility between switch technologies and signal types. The notation '?,' denoting 'possibly compatible,' indicates that, under normal operational circumstances, an automated routing algorithm would not attempt to make a connection of this type, though in an emergency such connections might be made in the absence of more appropriate infrastructure. Normally, signals would be prioritised, so critical signals would almost certainly be routed, but less important connections may be degraded or even omitted. For example, a non-critical redundant temperature sensor might be disconnected in favour of keeping an instrument package in operation.

	FPGA	FPTA	Digital X-bar	Analogue X-bar	MEMS	Relays	MOSFET/IGBT
Power	×	×	×	×	?	√	√
Heavy current analogue	×	×	×	×	?	√	√
Low current, low speed analogue	×	√	×	√	√	√	?
Low current, high speed analogue	×	√	×	√	√	?	?
Low speed digital	√	√	√	√	√	√	√
High speed digital	√	?	√	?	√	?	×
Low power microwave	×	×	×	×	√	×	×
High power microwave	×	×	×	×	?	×	×

× – Not compatible ? – Possibly compatible √ – Compatible

Table 1: Compatibility between switch technologies and signal types

2.4 Routing architectures

The major alternative switching architectures that may be considered when designing a reconfigurable manifold are as follows:

Crossbar Switch An $M \times N$ grid of switches configured to provide a M-input, N-output routing network.

Permutation Network A *permutation network* performs an arbitrary permutation on N inputs, such that any possible reordering of the inputs is supported.

Ad-Hoc and Hybrid Approaches Practical considerations make it appropriate to consider the possibility of leveraging existing COTS technologies, possibly in combination, to create reconfigurable manifolds. Though the result network topology and routing algorithms may be technically inferior to a purer design, economic considerations are nevertheless important for practical designs.

Embedding into Networks of Arbitrary Topology Given a sufficiently large and complex graph, with nodes representing switches and edges representing wires, it is possible to compute a switch configuration that implements an arbitrary circuit.

Each approach is described in detail below.

2.4.1 Crossbar switches

Crossbar switches have a long history, having originally been introduced as a means of routing telephone calls through electromechanical telephone exchanges. Conceptually extremely simple, a crossbar switch is constructed from two sets of orthogonal wires (bus bars in telecommunications nomenclature), such that each crossing can be bridged by a switch. Fig. 9 depicts the circuit of a small 8×8 crossbar switch.

To route a particular input to a given output, all that is necessary is for the switch corresponding to that input and output to be closed. Crossbar switches are somewhat inefficient in terms of hardware requirements, and also in terms of providing more routing capability than is strictly necessary in many cases – it is possible, for example, to route a single input to any number of outputs, or to common inputs together. Achieving reliability is relatively straightforward, however – replacing each non-redundant switch (Fig. 10) with a partially- or fully-redundant alternative (Fig. 11 or Fig. 12 respectively) allows single point failures to be recovered. A fully-redundant switch configuration allows any of its four component switches to fail-open or fail-closed without affecting functionality. The partially redundant version only requires half as many switches, but is only safe against fail-closed faults – however, given one or more spare bus bars on each axis, fail-open faults can easily be patched around and are therefore still recoverable. In cost terms, building a fully-redundant $M \times N$ switch requires $4 \times M \times N$ switches, whereas the partially redundant approach requires $2 \times (M+1) \times (N+1)$ switches, though clearly the larger circuit is more fault-tolerant. Though both circuits can accommodate at least one fail-closed fault per cross point, the smaller circuit is limited to only one fail-open fault across the entire switch for each additional pair of redundant bus bars.

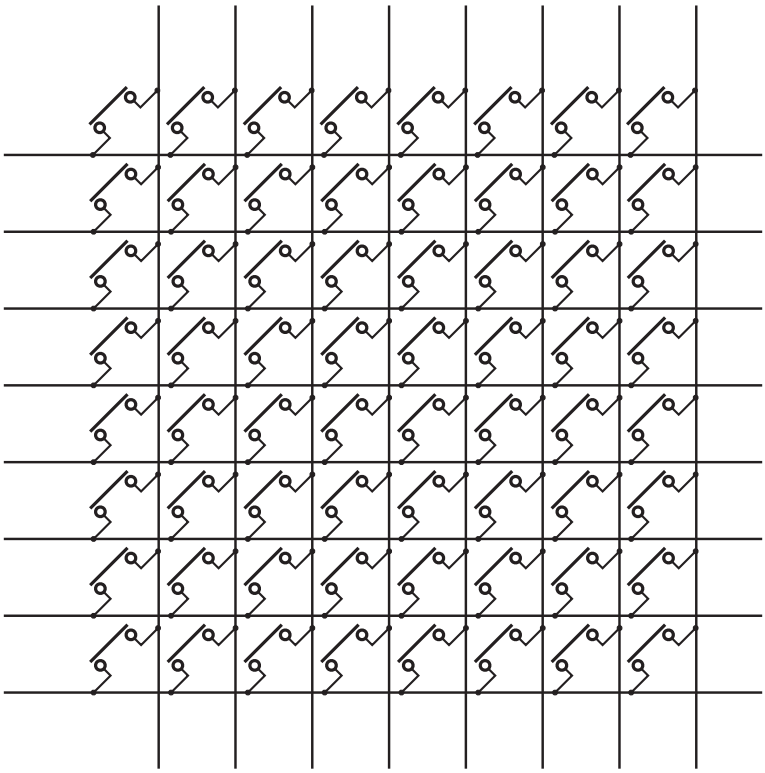

Figure 9: Crossbar switch

Figure 10: Non-redundant switch

Figure 11: Partially redundant switch configuration

A related architecture, once commonly used in circuit-switched telephone exchanges prior to the widespread introduction of digital technology, was the *Clos network* [14], which was normally constructed from three layers of smaller crossbar switches. This approach may or may not support all possible permutations depending upon the details of its construction – non-blocking Clos networks may provide an efficient means of building large manifolds from small crossbar ASICs, though from-scratch designs based on permutation networks (see below) are still likely to require fewer switches.

2.4.2 Permutation networks

Permutation networks are an alternative approach to routing that, in many cases, require substantially fewer switches for a given number of inputs – rather than $O(N^2)$, they tend toward $O(N \log N)$, which can be a very significant advantage when the number of inputs

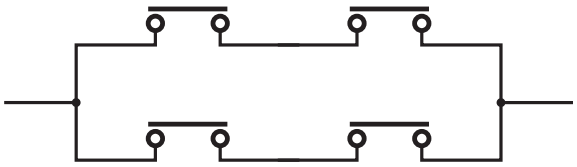

Figure 12: Fully-redundant switch configuration

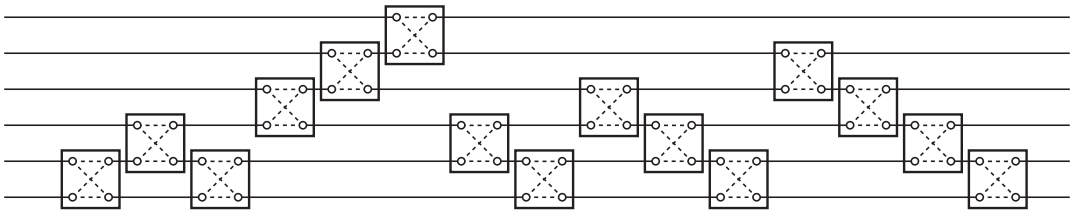

Figure 13: 6-way permutation network

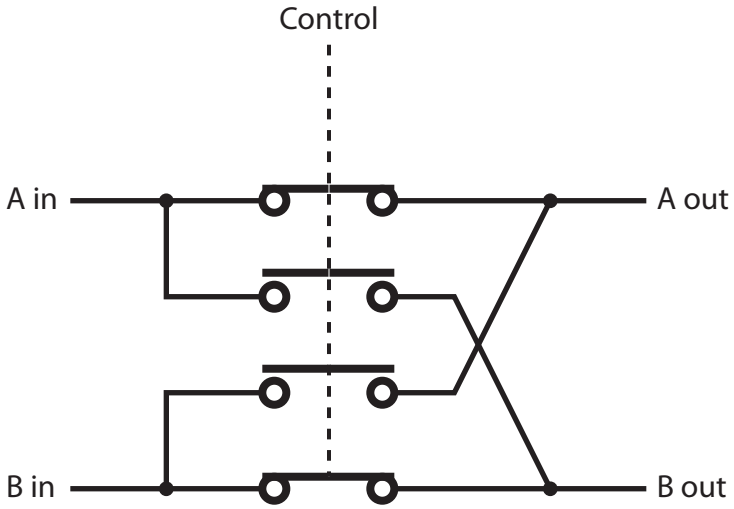

Figure 14: Swap node circuit

is large. Fig. 13 illustrates the concept with a 6-way permutation network. Its 15 swap nodes, each of which typically constructed from four switches (see Fig. 14), can each be in either of two states: pass the inputs left to right unchanged, or swap them. For 6 inputs, a crossbar switch is likely to be cheaper, in that it is likely to require only 36 switches, in comparison with 60 for the permutation network shown in Fig. 13. However, for 1000 inputs, assuming $N \log_2 N$, approximately $40,000$ switches are required, whereas a 1000×1000 crossbar switch would require 1 million switches.

Designing a permutation network can be somewhat baroque, though a useful relationship with *sorting networks* can be exploited. A sorting network [5, 6, 12, 15, 23] is architecturally similar to a permutation network, with the exception that the swap/don't swap decision at each node is made by comparing its inputs, such that its outputs are constrained always to respect a given ordering relation. Many well-known sort algo-

rithms, e.g. merge sort, bubble sort, transposition sort, bitonic sort or shell sort, can be constructed as sort networks. Since a sort may also be seen as just a particular kind of permutation, sort networks – by definition – must be capable of performing permutations. Furthermore, since the data to be sorted might initially be in any order, a sort network must be capable of supporting *all* possible permutations – therefore, if a sort algorithm can be adapted to create a sort network of arbitrary dimension, it follows that an equivalently structured permutation network would also be capable of any possible permutation. Usefully, the underlying sort algorithm can be leveraged to efficiently generate switch configurations, as follows:

1. Let $\langle W, < \rangle$ be a totally ordered set such that $|W|$ is the number of wires in the switch network, and each $w \in W$ represents exactly one input and one output.
2. Let the total bijection $P : W \rightarrow W$ represent the desired permutation to be implemented by the switch network.
3. Sort P with the underlying sort network, such that for each $(a, b) \in P$, a represents the input, and b represents the output. This can be achieved trivially by feeding tuples into the network ordered on a, then having the network sort these tuples ordered on b.
4. Note whether each swap node passed its data through unchanged, or whether it performed a swap. This gives the switch configuration for an isomorphic permutation network that performs an equivalent permutation.

Since suitable sort algorithms exist that have $O(N \log N)$ time complexity, computing a switch plan is therefore also an $O(N \log N)$ operation.

Permutation networks are nevertheless not always a better solution than crossbar switches, particularly when constructed as ASICs – their complex wiring reduces the effective advantage of their reduced switch count, particularly when considering that regular grids (crossbar switches being a particularly ideal example) are cheap and easy to lay out in comparison with the more spaghetti-like nature of large permutation networks, though Claessen *et al* [12] have shown promising results by adopting a *layout combinator* approach. Limitations on chip packaging limit the number of wires that can be physically connected to a single chip, which places hard limits on the impact of the $O(N^2)$ complexity problem with crossbar switches. However, when switches are large and/or expensive, as is the case with MEMS relays or any discrete component approach (e.g. full-size relays, MOSFETs, IGBTs), the reduction in component count could prove important.

2.4.3 Shuffle networks

Shuffle networks are essentially degenerate, incomplete permutation networks that do not support all possible permutations. Shuffle networks implement a *perfect shuffle,* [32], which typically allow any input to be routed to any output through $\log_2 N$ layers of switches. They are perhaps best known in the parallel computing world, where they are commonly used as high speed inter-processor interconnect architectures. Omega networks [24], a commonly used shuffle network architecture, typically require some kind of blocking or queueing hardware at each swap node so that collisions can be arbitrated.

In general, the incompleteness inherent in a single shuffle network is not tolerable for our application – it was, however, conjectured by Beneš in 1975 [7] and again more recently by Mary Sheeran [29] that exactly two shuffle networks in series can implement any possible permutation. The conjecture was recently proven by Çam [11], which means that this approach may lead to a means of designing compact, geometrically regular permutation networks that preserve $O(N \log N)$ complexity.

2.4.4 Ad-hoc COTS approaches

In some cases, COTS devices may be used to implement routing fabric. FPGAs, in particular, are ubiquitous, low cost and can be used (with appropriate considerations) in high radiation environments. There are a number of potential approaches:

1. Implement a general purpose crosspoint switch or permutation network as a HDL model, then synthesise it.

2. Generate HDL that routes the FPGA's inputs and outputs according to the desired switching plan, then synthesise the design.

The first option clearly limits the size of switch that can be implemented in a particular FPGA, though is inherently general purpose and can be reconfigured very rapidly. The second option is probably infeasible for embedded use at the time of writing due to the requirement for a complete tool chain in order to perform reconfiguration, though this situation may improve as technology supporting dynamic reconfiguration matures. In particular, the Xilinx jBits library [18, 30] allows FPGA configuration bitstreams to be generated on-the-fly from Java code, though it is currently unclear whether it can be feasibly implemented on the kinds of low-performance radiation-hard CPUs that are typically used for spacecraft applications.

2.4.5 Embedding into networks of arbitrary topology

A reconfigurable manifold of arbitrary topology may be represented by a graph whose nodes represent switches and whose edges represent wires. Embedding a desired circuit into such a network is essentially equivalent to computing a switch configuration. For the general case, this is a difficult computational problem that seems almost certainly to be in NP, with complexity rising exponentially with the number of switches in the network. Though this approach ultimately encompasses all others, in that both crossbar switches and permutation networks may be seen as special cases, the difficulty of computing switching plans makes it unlikely that this approach could be feasible in practice.

2.5 Make-before-break switching

At the device level, make-before-break switching requires the capability to establish a new connection, in parallel, before an old connection is disconnected. Where a reconfigurable manifold is routing signals that should not be temporarily interrupted, make-before-break switching allows a connection to be moved to an alternative route transparently to the signal's endpoints. In a reconfigurable manifold that does not alter its

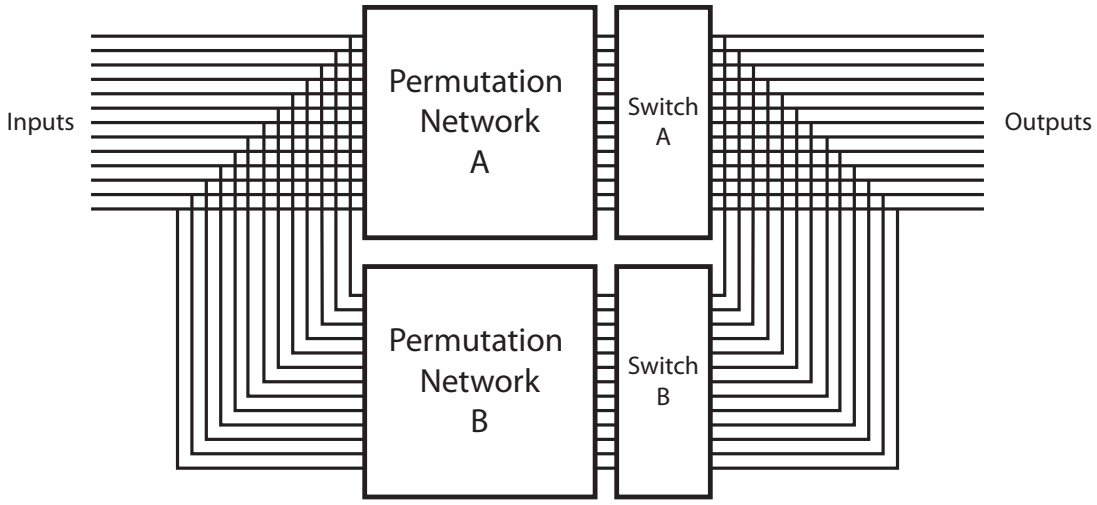

Figure 15: Work-around for make-before-break using permutation networks

wiring plan after it has been initially configured, support for make-before-break switching is unnecessary – however, such a capability is essential in order to support continuous automated testing and fault recovery (see Section 4).

Power, heavy current analogue, low-speed digital and low-speed analogue signals are all well suited to make-before-break switching, in that they are not particularly sensitive to minor changes in end-to-end resistance or discontinuities in impedance. However, high-speed digital, high-speed analogue, or (particularly) microwave signals need more careful consideration – in such cases, it may be necessary for the subsystems concerned to become involved in the routing process, at least from the point of view of being able to request that the manifold should not re-route particular signals during critical periods.

Crossbar switches support make-before-break switching by default: it is just necessary to turn on the switch for the new connection, waiting long enough (if necessary) for the switch to close fully and stop bouncing, then turn off the switch for the old connection. Implementing make-before-break switching in a permutation network is not feasible in general – making a change to a single route often requires several signals to be rerouted at once. A work-around solution is shown in Fig 15, where a pair of identical permutation networks is connected in parallel and are switched as follows:

1. Initially, Switch A is off and switch B is on.

   Permutation Network A is carrying all signals and Permutation Network B is not connected.

2. A new switch configuration is computed, and used to initialise Permutation Network B

3. Turn on Switch B.

4. Turn off Switch A.

   At this point, Permutation network B is now carrying all signals, and Permutation Network A is not connected.

For the next cycle of reconfiguration, the procedure continues with A and B swapped. This approach avoids switching glitches during reconfiguration of the permutation networks because whenever reconfiguration occurs, the permutation network in question is disconnected – actual switching of live signals only occurs during steps 3 and 4, which can trivially be arranged to be guaranteed clean.

Though this work-around implies slightly more than a doubling of hardware requirements, it nevertheless maintains $O(N \log N)$ complexity. Adding a third, redundant, permutation network as a hot spare allows modular redundancy to be implemented with a 3 times multiplier on hardware requirements, which compares well with the 4 times multiplier that would result from replacing each component switch with a redundant series-parallel switch network (see Fig. 12).

2.6 Grounding

Grounding of electronic systems within satellites is broadly similar to the grounding of Earth-based electronics; as-such, the same techniques and best practice applies in both cases. In satellites, grounding is particularly important because of the *charging effect,* whereby charged particles impacting the spacecraft impart a (potentially large) electric charge – careful grounding all conductive parts typically reduces or eliminates any consequential problems.

It is normal practice for a spacecraft to implement a ground network with a star topology – a single central grounding point is connected radially to the grounds on all subsystems. Cycles in the ground network are avoided, because they can form unwanted single-turn secondaries that may pick up hum or other unwanted noise from any heavy current subsystems in the vicinity.

Normally, grounds should not need to be switched by a reconfigurable manifold – a conventional, fixed, star ground topology should be sufficient for nearly all cases. Signals that are routed along shielded paths may require switchable ground connections[3] at one or both ends in order to avoid ground loops, though careful consideration of possible ground routing requirements may avoid this.

3 Self-organisation

In some circumstances, it is undesirable or even impossible to precalculate routing for a reconfigurable manifold. The responsive space paradigm requires that disparate subsystems should be able to be plugged together in any convenient manner, at which point they should self-organise and work together without human intervention. Achieving concept-to-launch times of the order of one week does not leave much time for anything other than physical assembly of the spacecraft, so the electronic subsystems must, of absolute necessity, not require a lengthy design process.

Self-organisation, at a fundamental level, requires subsystems to be able to discover each other, negotiate and configure any necessary wiring, and also to cooperate in maintaining the long-term reliability of the connectivity. These issues are discussed in detail in the remainder of this section.

[3]Also known as *ground lifts* to electrical engineers.

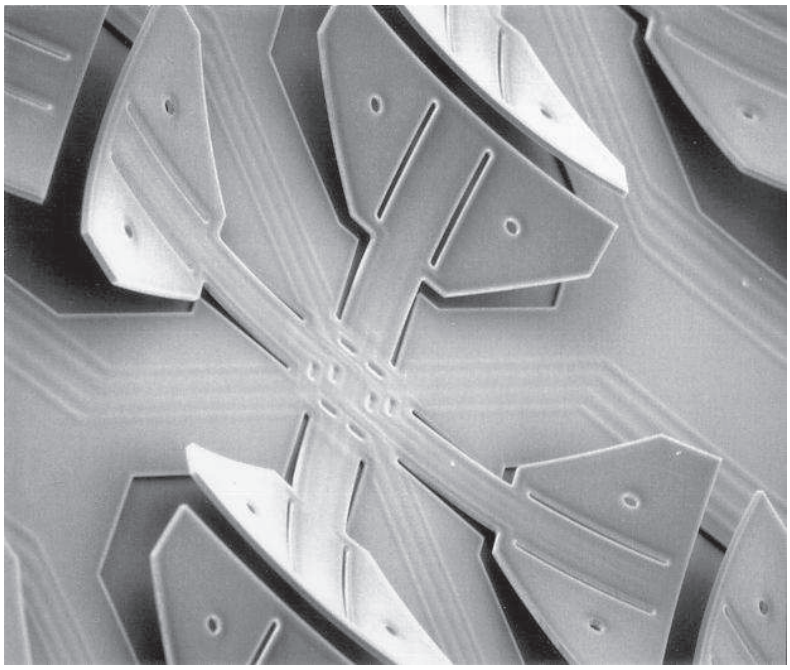

Photo: John Suh, University of Washington

Figure 16: Microcilia cell

3.1 'Space Velcro'

Some technologies absolutely require self-organisation in order to function at all. Fig. 16 is an electron micrograph of Joshi *et al*'s *Microcilia* concept [22, 33, 8]. MEMS technology is used to construct micron scale, articulated 'cilia' that are capable of manipulating small objects and of allowing the docking of small microsatellites. Assuming that electrical connections between the mated surfaces can be achieved, a self-organising, reconfigurable manifold based satellite could automatically configure any necessary connections during docking, then automatically recover the routing resources once the microsatellite has undocked.

Brei *et al* have investigated a passive interconnect architecture known as *Active Velcro* [13, 10, 9]. Fig. 17 illustrates the concept[4]. Mating, Velcro-like surfaces also contain a (possibly large) number of connectors, a proportion of which happen to make valid connections. Discovering these connections, then routing them via a reconfigurable manifold, potentially allows extremely straightforward ad-hoc construction. In manned spaceflight applications, an astronaut could connect or disconnect a piece of equipment simply by sticking or unsticking it to a Velcro-like pad[5]. In satellite applications, assuming that launch G force and vibration constraints are met, the same approach could allow extremely rapid construction and deployment.

[4]Note that this is the author's rendering, and is intended to be representational of the connectivity approach rather than an accurate physical description.

[5]It has long been standard practice to use Velcro to prevent small objects from floating around the cabins of manned spacecraft.

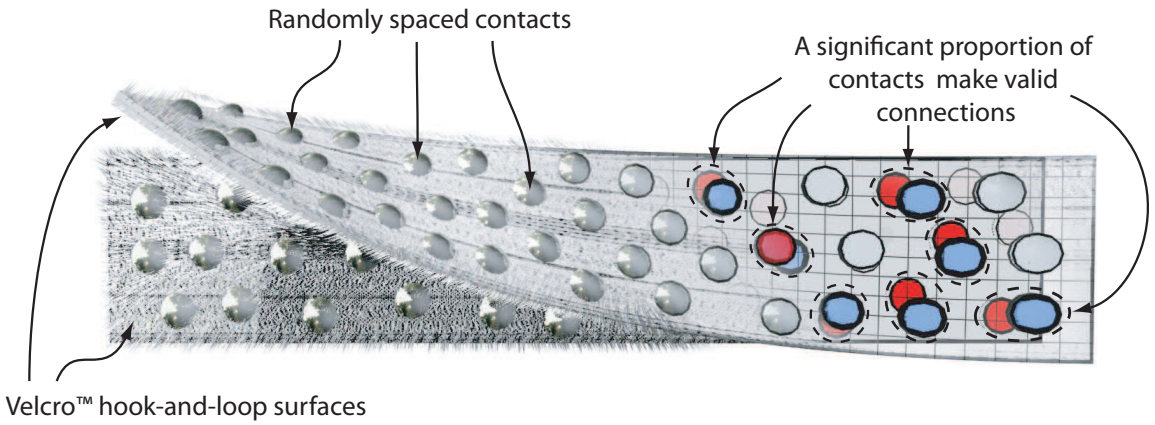

Figure 17: Active Velcro

3.2 Local routing

In a very small satellite, or within a single subsystem of a more complex satellite, routing may be exclusively *local*, i.e. switched only by a single level of switch networks. All connections in such a case would occur only to the edge of a single manifold, or cluster of sub-manifolds configured to act logically as a single manifold, with the consequence that the routing of all signals is equivalent only to routing across the manifold itself. Computing switch assignments for such an architecture is relatively trivial, with complexity of the order of $O(N^2)$ for a crossbar architecture or $O(N \log N)$ for a permutation network.

3.3 System level routing

Purely local routing requires a strict star architecture, with the manifold at the hub. This physical geometry does not suit all applications – in many cases, particularly in larger spacecraft, it is likely to be more appropriate to distribute the switching around the craft. Though it is theoretically possible to construct a large crossbar switch by ganging together smaller switches, this would be an expensive approach since the amount of inter-switch cabling would rise in proportion to the square of the number of switches. A more sensible and practical approach would be to construct a manifold-of-manifolds with an architecture resembling that of a circuit-switched telephone network – a number of manifolds handle primarily local connections internally, whilst handing off longer-distance connections via multicore trunk connections to other manifolds.

Computationally, the system level routing problem tends towards NP in the worst case (e.g. a manifold-of-manifolds where each manifold consists of exactly one switch and connectivity between manifolds is arbitrary is essentially the same problem that is discussed in Section 2.4.5), though the relatively small number of manifolds and relatively large amount of connectivity within each manifold is likely to minimise the consequences of this.

As with circuit-switched telephone networks, in general the manifold-of-manifolds approach would not support all possible permutations, which suggests that in responsive space applications, it makes sense either to adopt a local-routing-only approach, or to

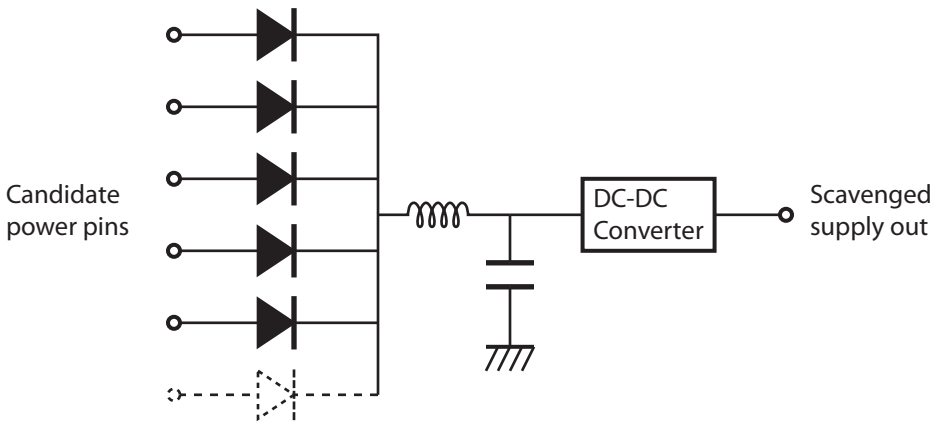

Figure 18: Power scavenging circuit

deliberately overspecify the amount of manifold-to-manifold interconnection resources.

3.4 Dynamic discovery

The *dynamic discovery* of connections is something that is becoming increasingly common in general-purpose computing. The USB standard, for example, allows devices to be discovered and configured automatically without significant human intervention. From the point of view of reconfigurable manifolds, the dynamic discovery problem is somewhat trickier, in that it is necessary to first power up any neighbouring subsystems, establish contact with them (potentially with zero prior knowledge of their wiring configuration), negotiate any required connections, then route the necessary signals. As a second requirement, it is then necessary to continuously re-test the existing connectivity in order that faults can be corrected and that subsystems coming on line or going off line can be connected and disconnected correctly.

In this section, the requirements for achieving reliable dynamic discovery, continuous testing and fault recovery are discussed.

3.4.1 The 'chicken-and-egg' initial power-up problem

It is a truism that any automatic discovery algorithm can only possibly run on hardware that is itself powered up. However, if a subsystem's power connections have not yet been discovered and configured, it will not (yet) be powered up – hence there is a chicken-and-egg problem. Though no longer in common use, a well-known solution already exists. For many years, the most commonly used PC peripheral interface standards, RS232 and Centronics, both suffered from a design oversight – no power supply pins – that proved maddening for any hardware engineer attempting to design small peripherals without separate mains power supply connections. Designers nevertheless succeeded in working around the limitation by including circuits that scavenged power from the I/O pins themselves. The technique is illustrated in Fig. 18 – a diode network, effectively a large-scale generalisation of a full-wave rectifier circuit, synthesises power rails effectively by implementing a minimum/maximum function on the voltages that are present. The clamping,

smoothing and DC-DC converter circuitry takes the potentially rather unpredictable raw output from the diode network and turns it into clean power that can be safely used to power up discovery circuitry prior to permanent routes being put in place.

Given suitable power scavenging circuits, a feasible power-up procedure for a large, manifold-of-manifolds architecture might be follows:

1. Power is applied to the first manifold through an arbitrary power pin.
2. The power scavenger circuit synthesises a suitable voltage rail for the embedded processor and discovery hardware responsible for the manifold.
3. All switches within the manifold are initialised to open circuit.
4. The power connection is detected, then connected via the manifold, thereby disabling the diode network. This step avoids the inherent voltage drop across the diode network, whilst also reducing power consumption and heat dissipation slightly.
5. The manifold starts to listen for connection requests from other subsystems (see Section 3.4.3).
6. Power is temporarily routed to arbitrary pins on neighbouring subsystems that currently do not appear to be active, giving them the chance to power up and begin their own discovery process. They may request that power is supplied through a different pin, if necessary, or request that the existing pin should remain connected indefinitely[6].

Eventually, all subsystems will be powered up, with the discovery process continuing to bring online all other necessary connections.

3.4.2 Watchdogs

It is standard practice for embedded processors in high reliability, mission critical and safety critical systems to be equipped with *watchdog circuits,* see Fig. 19.

A watchdog circuit is essentially a simple timer that is periodically reset by the host processor in such a way that, if the host processor happens to fail to reset it within a predetermined interval, the watchdog timer performs a hard reset on the host processor. Generally, this is integrated into a critical loop within the embedded software, so that if the program crashes the timer will fail to be reset, causing an automatic restart of the processor.

At a simplistic level, there is no reason why such a restart should cause problems for a manifold-of-manifolds architecture, though careful attention must be given to the following issues:

1. In the event of a watchdog reset, all external connections must be torn down, just in case the crash was itself caused by a faulty connection or, for example, by a single-event effect affecting the manifold itself.
2. Any negotiation protocol must be able to cope, e.g. by implementing timeouts, with connections going down without any corresponding explicit notification.

[6]Though it may be subject to change as part of the self-test algorithm.

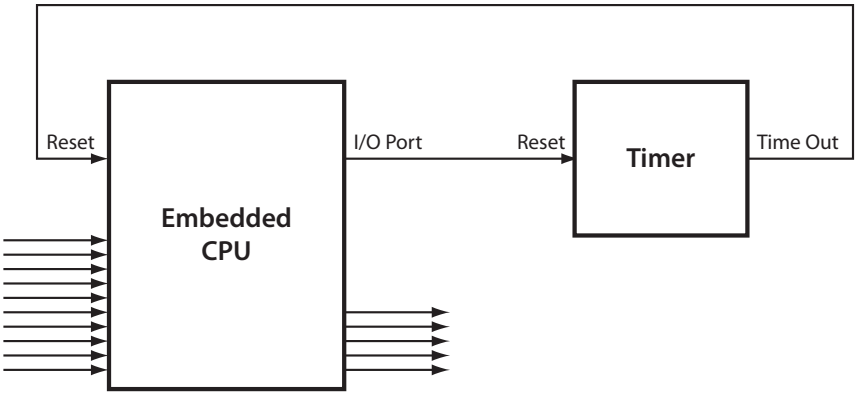

Figure 19: Typical watchdog circuit

3.4.3 Discovery probe circuits

Connection discovery depends upon an ability to safely probe connections to find out what neighbouring subsystem they are connected to. The outline circuit shown in Fig. 20 shows how a suitable 'discovery probe' might be implemented. The circuit shows a UART (bidirectional serial interface) connected to a host processor, whose serial I/O ports (marked TxD and RxD) assume good quality, logic-level signals. On the transmit side, the signal is first buffered in order to protect the UART[7], then high pass filtered to achieve AC coupling and connected to the probe output via a resistor, whose value should be carefully selected in order to limit worst case current in the event of an accidental connection to a power or high current analogue signal to a level that cannot cause damage. On the receive side, a similar current limiting resistor and high-pass network protects the active components from direct connection to otherwise potentially damaging signals. A DC-coupled linear amplifier boosts the signal, then a Schmitt trigger [28] (comparator with hysteresis) squares up the signal and raises it to logic levels suitable for the RxD input of the UART. Current limiting resistors should be chosen with values that are not too overspecified, since lower values are likely to result in better noise performance and higher achievable data rates.

In essence, the probe circuit is a simplified, extremely robust variation of a shared bus CSMA/CD network interface, in the style of 10Base2 Ethernet. AC coupling and a relatively high series resistance minimises the chance of damage due to accidental connection to higher voltage signals, whilst the ability to send and receive digital data without needing to switch between transmit and receive modes makes implementing higher level protocols relatively straightforward.

Sending NRZ (non-return to zero) serial data across AC coupled connections requires careful design of the low-level line protocol. Sending, for example, a long string of ones will cause the voltage to decay back to a centre value over a period of time that is determined by the time constant of the high-pass filter. Similarly, a data packet that consists predominantly of ones (or zeros) will tend to shift away from the most common value, causing an unwanted DC bias and consequential reduction in noise margins. Typically

[7]A high current buffer amplifier constructed from relatively large geometry transistors is far less likely to be damaged by a voltage spike than a UART I/O pin.

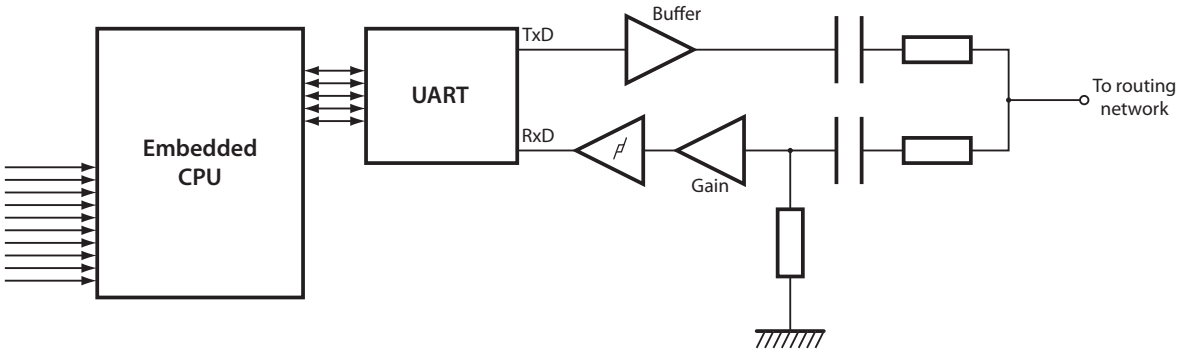

Figure 20: A possible discovery probe circuit

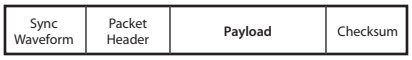

Figure 21: Typical packet format

this is addressed by arranging for the data encoding to implicitly retain an equal number of 0s and 1s – a trivial, though inefficient, approach is to spread an 8 bit byte across 16 bits, where each input bit corresponds to an inverted and a non-inverted copy in the output word. More efficient encodings exist that spread 2 bytes across 24 bits.

3.4.4 Line protocol

The main function of a suitable line protocol is to allow the discovery of connections, then to allow routing negotiation for signals. Probe circuits will typically alternate between sending packets that announce the identity of the relevant wire and listening for incoming packets that identify the other side of the connection. A suitable packet format is likely to follow the pattern shown in Fig. 21. Initially, a synchronisation waveform begins the transmission, whose purpose is to overcome any DC bias, whilst allowing the receiving UART time to lock on to the data. A packet header follows, identifying the kind of packet that is being sent, followed by the packet payload and finally a checksum.

3.4.5 Connection establishment

Connections are established as follows (assuming a single manifold):

1. Both endpoints announce their identity, and announce the identifier of the signal that they wish to connect to.
2. Manifold detects the announcements.
3. Manifold replies to both end points to say that the connection is being established, then ceases to probe either connection.
4. Manifold establishes the connection, within a predetermined maximum time interval.

5. Both endpoints are now free to use the connection.

More complex manifold-of-manifolds architectures will require more complex negotiation and routing, though the necessary protocols are likely to remain similar. A typical connection establishment protocol across a manifold-of-manifolds architecture would follow the following pattern (assuming that the endpoints are on different manifolds):

1. Both endpoints announce their identity to their local manifolds, along with a globally unique signal identifier.

2. Each manifold announces the signal's availability to neighbouring manifolds, along with a distance measure. This takes place separately for each endpoint.

3. Signal availability information continues to propagate across the manifold-of-manifolds. If a manifold receives connectivity information from more than one neighbouring manifold, this is ranked with the lowest distance measure first. Termination may be guaranteed by ensuring that connectivity information is propagated only when shorter distance measures are found than any previous measure announced on the same connection – the algorithm will therefore be guaranteed to reach a fixed point after at most a number of steps bounded by the number of manifolds in the system.

4. After a delay to allow propagation to complete, the endpoint manifolds now may use the routing information that has been collected in order to establish a shortest route across the manifold-of-manifolds.

This algorithm is essentially a distributed variation of Dijkstra's algorithm [16]. Since each manifold (graph node) has its own processor, time complexity is effectively $O(N)$ rather than the more usual $O(N^2)$, since processing power scales with N. The approach resembles the OSPF (Open Shortest Path First) routing protocol [26] in some respects.

Though this approach is relatively expensive in terms of the communications bandwidth required for routing, it nevertheless is unaffected by a dynamically changing architecture, or by component manifolds being unavailable, since routes will always be discovered if they exist, however complex they may be. In a fixed architecture satellite, some of this overhead may be avoided by precomputing the routing tables – such an approach would in principle be closer to the approach taken by the BGP (Border Gateway Protocol) [27].

3.4.6 Stale connection tear-down

In the event that a subsystem crashes, stale connections should be torn down after a known time-out interval. The discovery probe protocol should also allow a connection to be torn down more rapidly by announcing that a neighbouring connection is no longer in use. Assuming that a dynamic testing and fault recovery process will be continuously applied, there is no requirement for a 'keep alive' protocol to ensure that valid connections stay up (see also Section 4).

4 Dynamic testing and fault recovery

The same probe architecture necessary for discovery is also well suited to end-to-end testing of connections – if a connection is faulty (e.g. open circuit, shorted to ground or shorted to power), it will not be used, since the discovery process will fail to recognise it. As a consequence of this, at least for a short time after the discovery process has completed, all discovered connections may be regarded as functioning correctly. Over time, there is an increasing probability that, for example, permanent latch-up damage to a digital crossbar switch, may cause one or more connections to fail. This limitation can be avoided by constantly re-testing connections, ideally such that no connection may be established for a period longer than the minimum necessary to achieve the desired level of reliability.

4.1 Fault recovery protocol

There is actually no specific requirement to implement a fault recovery protocol as-such; the ability to set up and tear down connections, with make-before-break capabilities, is sufficient. Each end-point manifold should implement the following procedure (discovery and initial establishment of connections is assumed to have happened already):

1. Choose a signal on a round-robin basis.
2. Establish a second route to the same remote end-point through the discovery protocol, which has the side-effect of ensuring that end-to-end connectivity is currently valid.
3. Connect the signal to the newly established route, at both ends, whilst leaving the original connection in place.
4. Tear down the original connection.
5. Repeat.

Note that in larger systems, connections between manifolds must always provide sufficient spare connections to allow the discovery protocol to remain in operation at all times.

The stale connection timeout (see Section 3.4.6) should be longer than the worst-case time necessary to cycle through all connections.

When a connection fails, it will be repaired automatically the next time that the fault recovery procedure cycles through the relevant signal, because the failed route will no longer be detected, so it will naturally fall out of the pool of available connections.

4.2 Graceful degradation

In a situation where cumulative failures have exceeded the number of available connections, it is sensible to define a graceful degradation strategy in order to maximise the spacecraft's remaining functionality. A simple approach is to rank all signals in order of

importance, with signals toward the end of the list simply being disconnected if insufficient connectivity is available, though more sophisticated approaches may allow greater levels of recovery:

Routing signals on a less-ideal sub-manifold Normally, for example, digital data would be routed through dedicated digital switch networks. In the event that insufficient digital switching capacity remains, it is potentially feasible to route signals through spare capacity in other switch networks, e.g. via MEMS switching that would normally be used for microwave signals or via high speed analogue routes.

Multiplexing Manifolds could potentially be equipped with multiplexing hardware, in order that multiple low speed signals could be routed through a single connection. Though this may degrade any signals carried in this way, it may still be preferable to disconnecting signals entirely.

Emergency backup routing As an extension to the multiplexing approach, in an emergency backup routes could be established by non-standard means, such as via low power local digital radio links.

5 Conclusions

At the time of writing, this technology is at a relatively early stage of development; nevertheless, it is possible to determine the following advantages of reconfigurable manifolds over conventional fixed-architecture spacecraft wiring harnesses:

Cost Reduction Since a reconfigurable manifold does not need to be designed from-scratch for each satellite, considerable cost reductions in terms of initial design, validation and verification are likely.

Reduction in Time To Launch (Responsive Space) Reduced design effort has a direct effect in terms of calendar time, potentially helping reduce a design process that is conventionally measured in years to just weeks or even days.

Possibility for Re-purposing After Launch If a spacecraft is no longer required for its initial purpose, given a modular design, it is quite likely that it could be re-purposed after launch at very low cost. For example, an imaging satellite with excess communications bandwidth could, assuming it has enough fuel, be shifted to another orbit to act as a communications relay.

Disaster Recovery Now legendary, the recovery of Apollo 13 after an explosion that deprived the command module of all three of its fuel cells and its entire oxygen reserve, with all crew alive and unhurt [34], was a direct consequence of heroic efforts to jury-rig the lunar lander's oxygen systems in order to keep the crew alive. A conventional satellite has no astronauts with a kit of spare parts available to make repairs – typically, failures tend to be terminal. A reconfigurable manifold offers great potential for jury-rigging the craft, either from Earth or possibly autonomously, so as to allow it to continue with some or all of its mission.

Mass reduction By sharing redundant wiring capacity across all subsystems, the total amount of copper necessary is reduced considerably in comparison with modular-redundant conventional wiring. At approximately $30,000 per kg to low earth orbit, even small savings can have considerable consequences in terms of cost.

The responsive space paradigm makes it essential for plug-and play concepts that are now ubiquitous in desktop computing (e.g. PCI [1], USB [2] and FireWire/IEEE 1394 [19, 20, 21]) to be migrated to satellite architectures. Though in some cases these technologies may be used directly (USB, in particular, is currently in use in satellites), digital networking alone is insufficient. The reconfigurable manifold approach, however, allows similar results to be achieved for almost all kinds of signal.

5.1 Future Work

Many, if not all of the prerequisites for the practical construction of satellites based upon reconfigurable manifold technology are well-established, so the problem is primarily one of systems integration rather than difficult original R&D. The next step we intend to take is to build a software simulation of a reconfigurable manifold in order to test the feasibility of the approach. Beyond that, given appropriate funding and the necessary political will, it just remains to design a practical implementation and, hopefully, to test it in space.

Acknowledgements

This work was supported by the US Air Force Office of Scientific Research Space Vehicles Directorate, through an EOARD grant. The first author wishes to thank AFOSR at Kirtland AFB, and Jim Lyke in particular, for their help and advice, without which this work would not have been possible.

Acknowledgement of Sponsorship

Effort sponsored by the Air Force Office of Scientific Research, Air Force Material Command, USAF, under grant number FA8655-06-1-3021. The U.S. Government is authorized to reproduce and distribute reprints for Government purpose notwithstanding any copyright notation thereon.

The views and conclusions contained herein are those of the author and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Office of Scientific Research or the U.S. Government.

References

[1] PCI special interest group web site. http://www.pcisig.com/.

[2] Universal serial bus specification. Revision 2.0, http://www.usb.org/, 2000.

[3] *AD8116 - 200 MHz, 16 $\times$ 16 Buffered Video Crosspoint Switch*. Analog Devices, 2006. http://www.analog.com/en/prod/0,2877,768

[4] *High Performance Crossbar Switch for Virtex-II and Virtex-II Pro FPGAs*. Xilinx, 2006. www.xilinx.com/esp/xbarswitch.htm.

[5] AJTAI, M., KOMLÓS, J., AND SZEMERÉDI, E. An O(n log n) sorting network. *Combinatorica 3(1)* (1983), 1–19.

[6] BATCHER, K. E. Sorting networks and their applications. In *Proc. AFIPS Spring Joint Computer Conference 32* (1968), pp. 307–314.

[7] BENEŠ, V. E. Proving the rearrangeability of connecting networks by group calculations. *Bell System Tech. J.*, 45 (1975), 421–434.

[8] BOHRINGER, K. F. A docking system for microsatellites based on microelectromechanical system actuator arrays. Tech. Rep. AFRL-VS-TR-2000-1099, US Air Force Research Laboratory, Space Vehicles Directorate, September 2000.

[9] BREI, D., AND CLEMENT, J. Proof-of-concept investigation of active velcro for smart attachment mechanisms. Tech. Rep. AFRL-VS-TR-2000-1097, US Air Force Research Laboratory, Space Vehicles Directorate, September 2000.

[10] BREI, D., AND CLEMENT, J. Velcro for smart attachment mechanisms. Tech. Rep. AFRL-VS-TR-2001-1104, US Air Force Research Laboratory, Space Vehicles Directorate, August 2001.

[11] ÇAM, H. Rearrangeability of $(2n-1)$-stage shuffle-exchange networks. *SIAM J. Comput. 32*, 3 (2003), 557–585.

[12] CLAESSEN, K., SHEERAN, M., AND SINGH, S. The design and verification of a sorter core. In *Proc. CHARME'01, LNCS 2144* (2001), Springer-Verlag.

[13] CLEMENT, J. W., AND BREI, D. E. Proof-of-concept investigation of Active Velcro for smart attachment mechanisms. In *In Proc. 42nd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference and Exhibit* (2001). AIAA Paper 2001-1503 (AIAA Accession number 25238).

[14] CLOS, C. A study of non-blocking switching networks. *Bell System Technical Journal 32*, 2 (1953), 406–424.

[15] CORMEN, T. H., LEISERSON, C. E., RIVEST, R. L., AND STEIN, C. Chapter 27: Sorting networks. In *Introduction to Algorithms*. MIT Press and McGraw-Hill, 1990, pp. 704–724.

[16] DIJKSTRA, E. W. A note on two problems in connexion with graphs. *Numerische Mathematik 1* (1959), 269–271.

[17] FOUST, J. Smallsats and standardization. *The Space Review* (2005).

[18] GUCCIONE, S. A., LEVI, D., AND SUNDARARAJAN, P. JBits: A Java-based interface for reconfigurable computing. In *Proc. 2nd Annual Military and Aerospace Applications of Programmable Devices and Technologies Conference (MAPLD)* (1999), NASA, available at http://klabs.org/mapld/index.htm.

[19] IEEE P1394 WORKING GROUP. *IEEE Std 1394-1995 High Performance Serial Bus*. IEEE, 1995.

[20] IEEE P1394A WORKING GROUP. *IEEE Std 1394a-2000 High Performance Serial Bus – Amendment 1*. IEEE, 2000.

[21] IEEE P1394B WORKING GROUP. *IEEE Std 1394b-2002 High Performance Serial Bus – Amendment 2*. IEEE, 2002.

[22] JOSHI, P. B. On-orbit asssembly of a universally interlocking modular spacecraft (7225-020). Tech. Rep. NASA SBIR 2003 Solicitation Proposal 03- II F5.03-8890, NASA, 2003.

[23] KNUTH, D. E. Section 5.3.4: Networks for sorting. In *The Art of Computer Programming, Volume 3: Sorting and Searching, Third Edition*. Addison-Wesley, 1997, pp. 219–247.

[24] LAWRIE, D. H. Access and alignment of data in an array processor. *IEEE Transactions on Computers 25* (1976), 1145–1155.

[25] LYKE, J., WILSON, W., AND CONTINO, P. MEMS-based reconfigurable manifold. In *Proc. MAPLD* (2005), NASA, available at http://klabs.org/mapld/index.htm.

[26] MOY, J. RFC 2328: OSPF Version 2. *IETF* (1998).

[27] REKHTER, Y., LI, T., AND HARES, S. RFC 4271: a Border Gateway Protocol 4 (BGP-4). *IETF* (2006).

[28] SCHMITT, O. H. A thermionic trigger. *Jour. Sci. Instr. 15*, 1 (1938), 24.

[29] SHEERAN, M. Puzzling permutations. In *Proc. Glasgow Functional Programming Workshop* (1996).

[30] SINGH, S., AND JAMES-ROXBY, P. Lava and JBits: From HDL to bitstream in seconds. In *Proc. 9th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM'01)* (2001).

[31] STOICA, A., ARSLAN, T., KEYMEULEN, D., DUONG, V., GUO, X., ZEBULUM, R., FERGUSON, I., AND DAUD, T. Evolutionary recovery of electronic circuits from radiation induced faults. In *Proc. IEEE Conference on Evolutionary Computation* (2004), IEEE Computer Society.

[32] STONE, H. S. Parallel processing with the perfect shuffle. *IEEE Transactions on Computers 20*, 6 (1975), 57–65.

[33] SUH, J. W., DARLING, R. B., BOHRINGER, K. F., DONALD, B., BALTES, H., AND KOVACS, G. T. A. SMOS integrated ciliary actuator array as a general-purpose micromanipulation tool for small objects. *IEEE Journal of Microelectromechanical Systems* (1999).

[34] TURNILL, R. *The Moonlandings: an eyewitness account*. Cambridge University Press, 2003.